DOCUMENT RESUME

ED 294 393 EC 202 564

AUTHOR Lindsay, Peter H.
TITLE Computers in Special Education: Using Technology To Make Up the Difference.
PUB DATE Mar 86
NOTE 27p.; In: The Proceedings of the Canadian Symposium on Special Education Issues (Toronto, Canada, March 4-5, 1986), see EC 202 558.
PUB TYPE Speeches/Conference Papers (150) -- Information Analyses (070)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS Computer Assisted Instruction; Computer Simulation; *Computer Uses in Education; Database Management Systems; *Disabilities; Drills (Practice); Educational Needs; Elementary Secondary Education; *Gifted; Hearing Impairments; Learning Strategies; Metacognition; *Microcomputers; Programing; *Special Education; *Technology; Tutoring; Visual Impairments; Voice Disorders; Word Processing
IDENTIFIERS Proofreading

ABSTRACT

This paper reviews the current role of microcomputers as either a cognitive or sensory prosthesis, and evaluates their potential for meeting the learning needs of exceptional students. In the area of declarative learning, computers are used for drill and practice, tutoring, and "intelligent' tutoring. In the area of procedural learning, computer-based learning environments involve computer simulation; programming languages; tools for manipulating language such as word processors and proofreading assistance programs; tools for manipulating knowledge such as database management systems; and programs that teach students how to reflect upon and begin to understand their own cognitive processes. Computers also have a unique role as a potential sensory prosthesis for hearing-impaired students, visually-impaired students and students with no voice. (JDD)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED294393

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

☑ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

COMPUTERS IN SPECIAL EDUCATION: USING TECHNOLOGY TO MAKE UP THE DIFFERENCE

Peter H. Lindsay, Ph.D.

The Ontario Institute for Studies in Education

Spring 1986

EC 203564

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

Barry Lee

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

COMPUTERS IN SPECIAL EDUCATION:
USING TECHNOLOGY TO MAKE UP THE DIFFERENCE

A. PROGRAMS FOR AUGMENTING THE MIND 1

I) Declarative Learning with computers 2
a) Drill and Practice 2
b) Tutoring 2
c) "Intelligent" tutoring 3

II) Procedural learning with computers 5

a) Computer simulation 5
b) Learning to program 7
c) Tools for manipulating language 10
i) Wordprocessors: The first generation 11
ii) Educational wordprocessors: What makes them different? 12
iii) Second generation wordprocessors 13
iii) Proofreading assistance 14
d) Tools for manipulating knowledge 16
e) Mirroring the mind 17

B. PROGRAMS FOR AUGMENTING THE SENSES 19

I) Computers for students who can't hear 20

II) Computers for the visually impaired 21

III) Computers for students with no voice 22

C. REFERENCES 23

In our modern information age, computers are everywhere. They are in our cars and our washing machines as well as our businesses and our banks. In recent years, with the advent of the microcomputer, computers are finally making their way into our schools in large numbers.

The bulk of computer use in our schools so far has been for teaching programming to regular students. More and more however, pioneering teachers are experimenting with the use of computers to teach cildren with exceptional learning needs. Some of these applications attempt to develop programs that can compensate for a child's difficulty with inputting or outputting information whether it is due to visual impairment, a hearing loss or perhaps with an inability to speak. Other applications focus on the exceptional child's cognitive impairments. The development and use of computers in both of these areas are beginning to have a impact on the way the exceptional child is taught. Hopefully it will ultimately also have an impact their ability to lead a productive independent life.

This paper reviews the current role of microcomputers as either a cognitive or sensory prostheses. Since cognitive prostheses are by far the most numerous, much of the discussion will concern these applications. In each case, the main characteristics of the approach will be outined then followed by an evaluation of its potential for meeting the learning needs of exceptional students.

PROGRAMS FOR AUGMENTING THE MIND

A large variety of different approaches to using computers with exceptional students have been proposed and attempted (Goldenberg, Russell, Carter, 1984; Hagen, 1984). To evaluate these approaches, it is useful to keep separate two different types of learning -- declarative learning and procedural learning. Declarative learning refers to the learning involved in the acquisition of the facts related to any topic - the acquisition of number facts in arithmetic, historical dates, the names of the capital cities in Canada and so forth. Procedural learning on the other hand, involves learning how to manipulate those facts in meaningful ways. It is concerned with the ability to see patterns in the information being presented, to manipulate and synthesize new relations from old ones, to be able to construct models in the information domain in which one is working in. In short, declarative learning refers to the acquisition of information; procedural learning refers to the acquisition of knowledge.

Declarative Learning with Computers

Drill and practice

Among the first situations in which computers were used in schools were learning situations in which the primary goal was to consolidate previously learned declarative information. Such consolidation has been traditionally achieved by frequent repetition, that is by drilling students until their responses became rapid and automatic. Drill and practice activities were obvious candidates for computerization and were among the first types of programs to become available for teachers in the classroom. It is important to keep in mind however that the drill and practice program does not attempt to actually "teach" the material. As the name implies, these programs are primarily used to practice a student on material that has already been taught by other means.

At present there are a large number of quite effective, reasonably priced programs for practicing students in all sorts of things. They are available in areas as diverse as basic arithmetic, spelling, and learning about prefixes and suffixes. In fact there are now a significant number of program packages that attempt to provide drill across the complete elementary school language arts and mathematics curriculum (e.g., see the Milliken Mathematics series and the Milliken Language Arts series).

When educators complain about the lack of suitable software for computers, they are usually reacting to the quality of the early drill and practice programs. These are among the easiest programs to implement and hence were among the first programs to be made available to teachers. Unfortunately much of the earlier work was produced by people whose main skills were in programming and computer technology rather than in pedagogy.

Tutoring

A second basic class of programs that are used for declarative learning are tutorial programs. They differ from drill and practice in that in addition to posing questions on the subject matter, tutorials contain a description or explanation of the concepts being taught. The basic format of a tutorial is to provide some instruction then ask questions to see if the student understands On the basis of the student's performance, further instruction is given. Tutorial programs are now also available for a wide range of topics

covering subject material as diverse as history, grammar, sentence construction, astronomy and even basic social and life skills.

In general, computers are extraordinarily well suited for providing declarative learning opportunities either in a drill and practice mode or in a tutorial mode. Computers can be programmed to be completely flexible in terms of the content, the rate, the pacing and the type of information presented to the student as well as the degree of learner control. They can provide instantaneous feedback to the learner. Hence the computer is ideal for developing a fully individualized and personalized declarative learning environment.

To this must be added the fact that unlike almost any other type of declarative learning situation, students seem actually to enjoy doing drills and tutorials when they can do them on a computer.

The individualization that is part of the computer based declarative learning environment is ideally matched to the special learning needs of the student with a general or specific learning disability. Students with these difficulties seem to require endless amounts of practice with most material before retrieval becomes automatic and fast enough to be functionally useful. Computers have great potential for providing the extensive practice that is necessary. In fact, in some situations, computers seem to be the only approach that has any chance of achieving success. This is not to say of course that teachers should attempt to make the computer carry the whole teaching burden for the exceptional student's declarative learning. It is only to point out that the computer can be used as a very effective teaching tool for this aspect of the exceptional learners' school experiences.

"Intelligent" tutoring

A relative newcomer to the tutorial scene is the so-called intelligent tutor or intelligent computer assisted instruction (ICAI) (Hayes-Roth & Thorndyke, 1985; Pea, 1985). Unlike its older brother, intelligent tutors do not have a fixed set of rules that are used for branching to present some particular material depending on the student's answer to a set of questions. The basic approach taken in an intelligent tutor is to try to mimic, as far as possible, the behaviour of a live tutor.

A typical intelligent tutoring program starts out by having a relatively natural language dialogue with the student. During this dialogue, the program attempts to determine just what the student's skill or knowledge level is in the domain to be taught. Once the student's knowledge level is established, the program decides on the best type of material and best mode of instruction for this particular student. The optional instructional mode include such things as hints towards a solution, sets of relevant problems for the student to try to solve that illustrates a particular point, appropriate "coaching tips" or perhaps some textual or pictorial information to supplement an explanation. The most appropriate material and instructional approach used at

any point is based on the program's "diagnosis" of the student's current learning state as well as the program's (and hence the programmer's) best guess as to the most appropriate way to move the student to the next higher level knowledge state. A properly implemented intelligent tutor then must have a large number of student "models" or representations of the various knowledge states that students typically have about a topic. These representations should include the most frequent misconceptions students have of the material as well as the correct conceptions. In addition, the program has sets of rules to guide it in selecting the specific types of assistance to use depending on the currently diagnosed knowledge state.

Prototype models for intelligent tutors are available now in quite a variety of subject areas. Some of them deal with what initially might appear to be quite simple tasks such as teaching basic addition and subtraction (Brown & Burton, 1978). Although the task seems simple, the program for teaching it isn't! This particular program has a very complete error analysis routine for detecting "bugs" in children's thinking about addition and subtraction that itself requires more than 150,000 lines of code.

Other intelligent tutoring programs have been written for more complex topics such as high school algebra and geometry (Brown, 1985), programming in LISP (Anderson & Reiser, 1985), and medical diagnosis (Clancey & Letsinger, 1981). While several of these programs such as Anderson & Reiser's are now available as products that can be used to teach students certain topics in a standard teaching environment (i.e., the LISP programming course at Carnegie Mellon University), for the most part, these program are still in the prototype stage. Moreover, they are large programs that usually require much more powerful computers to execute than are usually available in today's schools. The developers of these programs however are promising to have operationally effective and economically intelligent tutoring systems available for schools within five years.

Will these intelligent tutoring programs be more effective with exceptional learners than their dumber brothers have been? Like their predecessors, intelligent tutoring programs have at least the initial advantage of offering the student instruction that is totally personalized and a tutor whose attention is devoted solely to that single student.

The question is whether the knowledge states that an exceptional learner goes through in mastering some material is any different than what a normal student goes through in mastering the same material. Will the misconceptions the exceptional learner develops be different in some way and if so, how? The fact that they may be a different notion is not in itself a problem. The knowledge base of an intelligent tutor is built up gradually over time as a result of successive encounters with students. If exceptional students were included as part of the mix of experiences to which the tutor was exposed, then whatever idiosyncratic states there are associated with exceptional learners would be included.

Procedural Learning with Computers

There is little question that a very effective declarative learning environment can be built around computers for students of all ability levels. A much less certain issue is just how effective computers are in procedural learning situations. As mentioned before, procedural learning requires situations where the learner is able to actively manipulate information, to generate and test hypothesis and evaluate the results, and to explore the implications of synthesizing old elements in new ways. When we talk about declarative learning we usually talk about rote memorization. When we talk about procedural learning we talk about understanding, interpreting, and being able to make inferences.

The first attempts to provide computer based learning environments for procedural learning were of two basic types. The first type was the computer simulation. The second was the teaching of programming as a way of providing an opportunity for procedural learning. The third and most recent approach is to design computer based "cognitive tools" that can augment the various cognitive activities carried out in the performance of some intellectual task. The emergence of cognitive tools seems to have the most profound potential for altering both the way we teach as well as the kinds of things students learn. It has a special significance for the exceptional learner since these cognitive tools provide the possibility of being a cognitive prostheses.

Computer simulation

As the name implies, computer simulations are programs that mimic or simulate some real-life situation. Some of the more popular simulations are programs that simulate the operation of a simple business (e.g., a lemonade stand) or a country's economy. Others simulate more complex or perhaps dangerous situations like the operation of a nuclear power station during a threatened meltdown.

To get an idea of the types of activities the students engage in while working with one of these simulations, consider two recent simulations that seem to have substantial educational value. The first of them is an imaginative adventure style game designed to teach Canadian history. In this program, the student can play the role of great Canadian explorers as they explored Canada. Hence they can make decisions that lead to "getting lost" or being "attacked by Indians" or of "dying of scurvy".

The second simulation, **Carmen de Sandiego**, focuses on geography and social studies. In this program, the student tries to track down international criminals. On different adventures, the criminals flee to different parts of the world. The student follows the fugitive by gathering clues as to her whereabouts from hotels, airports and the Interpol offices in various countries. The cues however are not straightforward. They refer to such things as currencies of different countries and to ancient countries that are no longer on modern maps so that the student must look up the pertinent information on the encyclopedia or maps provided with the program in order to make sense of the clues. Thus, in addition to the geography it teaches, the program provides a motivating environment that encourages the student to learn how to look up information in various sources and put together in order to draw the proper inference.

Good educational simulations should have a number of features. One of them of course is realism. A second is the presentation of an active responsive learning environment that is motivating for the student to explore. The third and most important feature from a pedagogical point of view is that the simulation provide an opportunity for the student to come to an understanding of the system through a cycle of hypothesis proposing, testing then deciding on which hypothesis to try next on the basis of feedback received. It encourages systematic thinking. If one wants to gain full control of the simulation, one must systematically manipulate the variables and synthesize the results of those manipulations. On the other hand, it should not be too pedantic. Anyone should be able to "play" casually in a simulation environment and learn enough to find it interesting.

Recently developed simulations are becoming extraordinarily complex. They are complex both in terms of the area which is being simulated and in terms of the realism of the simulation. A particularly impressive example of the new genre of simulations is **STEAMER**, a simulation designed to train U.S. navy seamen in the complexities of starting up a steam boiler on a ship (Hollan, Hutchins, & Weitzman, 1984). An even more famous simulation program, **SOPHIE**, was developed earlier to teach electrical engineers about the intricacies of electrical circuits (Brown, Burton, & deKleer, 1982). These simulation programs demonstrate some of the more advanced techniques that we will probably see in the school simulations of tomorrow. At present however, they require far more computing power than is available in today's schools.

Which type of exceptional learner can profit most from the simulation approach to a computer based learning environment? The open endedness of a simulation environment and the emphasis on learner control are characteristics that would seem to make simulations particularly well suited to the learning style of gifted learners. Properly designed simulations however can be played at a number of different levels. Hence simulations are generally highly motivating, effective learning environments for student of all ability levels.

Learning to program

The second major approach to providing procedural learning opportunities in schools is through programming courses. The most frequent use of microcomputers in today's schools is for teaching programming. Moreover there is a great deal of rhetoric promoting the teaching of programming and programming courses are being increased at the expense of other things. Given these circumstances, it is important to consider carefully the justifications people present for teaching students to program.

Programming as a vocation: The early arguments for teaching programming to students were not based on the value of programming courses for teaching general thinking skills. Rather they were based on the notion that society is entering an information age in which it will be critical for all citizens to be computer "literate". Moreover it is the responsibility of the schools to make sure the students become computer literate just as it is the school's responsibility to teach them to read and write. According to some of the early computer zealots, computer literacy is not simply knowing about computers; it is knowing how to make them do what you want them to do.

Some also claimed that the programming skills being taught in school could be justified in terms of providing a useful vocational skill. It would open up potential job opportunities for the students in the coming information age. The same set of arguments applied to teacher training produced the claim that all regular classroom teachers should also be taught to program. After all, if teachers are going to be able to use computers effectively in their classroom, isn't it necessary for them to learn to program?

The short answer is "no"! As the potential role of computers has become better understood, it has become increasingly obvious that it is not necessary for either the teacher or the student to become a programmer in order to be able to make very effective use of a computer as a teaching and learning tool. The student does not have to know how to write a simulation program in order to learn from it anymore than the teacher has to know how to design or develop a wordprocessing package in order to be able to make very effective use of a wordprocessor in the classroom. Furthermore, the potential job opportunities in the programming field are not nearly so numerous as was once anticipated and the level of programming proficiency necessary for a commercially employable programmer is not achievable in a regular school course. Programming at a commercial level is a highly skilled, specialized talent that is unrealistic to expect a large proportion of the population to have.

In short, the early arguments for teaching programming either as a necessary part of a general course in computer literacy or as a job training activity are simply valid. This is not to say that no student should learn to program or that no programming courses should be offered in the schools. It is only to note that the reasons presented for offering programming in schools should

be carefully scrutinized.

Programming as a way of teaching thinking skills: If one cannot justify programming as vocational training or as a necessary part of a computer literacy course, can one justify it on the grounds that it provides an important procedural learning experience? Here the issues are much more interesting and important for the educator. The most imaginative and persuasive of the apologists for the use of programming to teach thinking skills is Seymour Papert.

Papert, an MIT physicist who studied with Jean Piaget, is the inventor of LOGO, the most widely used "educational" language in the schools. Papert has been the most provocative of the advocates for teaching children to program as a way of exposing them to a powerful and flexible procedural learning environment (see Papert's 1980 book, Mindstorms, for his most complete and eloquent statement of his point of view).

According to Papert and his many followers, when children program in LOGO, they are also learning many very important thinking and problem solving strategies. They spontaneously master many "powerful ideas" such as the importance of breaking up a complex problem into "mind sized bytes", of developing a logical, ordered sequence of steps in working towards a solution, of systematically testing one's hypotheses in order to detect and correct "bugs" in one's thinking. For Papert, the LOGO learning environment contrasts sharply with computer based drill or tutorials. The important distinction is who is in control of the learning: the child or the computer. LOGO supporters believe it is axiomatic that it is better to have the "child program the computer" than the "computer program the child".

In principle, any computer language, such as BASIC could be used to provide this type of procedural learning environment. LOGO however has some features that make it especially easy to use this way. What has been built into LOGO is the possibility of constructing a wide range of different "microworlds" or cognitive playgrounds for the child to explore. Although LOGO comes with a fixed set of primitive commands for manipulating text, graphics and music, its beauty and power is that it is so easy to redefine. With relatively little effort, teachers can define and build completely new sets of commands out of the basic primitives in order to construct a programming environment specifically suited to a particular student population or subject area. In that sense LOGO is a general purpose educational tool for constructing various types of learning environments suited to the needs of specific students.

Some concrete examples will make this feature of LOGO clearer. The commands used to change the background colour of the screen in LOGO is the screen command ("background") followed by a code number to indicate the desired colour (e.g., "background 4" changes the screen colour to blue). Suppose the teacher wishes to use the computer as a way to reinforce the student learning colour names. To do so, the teacher could simply relabel the "background 4" command to be "blue". From then on, whenever the student types "blue", the screen colour on the monitor would turn blue.

Similarly most LOGO primitives require the user to type in two pieces of information, a command (e.g., the "background" in the example above) and a

parameter or number value that provides additional information needed by the command (e.g., in this case the "4" to indicate the colour). This amount of typing however may not be suitable for the very young student or the student who has severe physical disabilities. In LOGO that poses no problem. Simply redefine the commands to be single keystrokes. Similarly the basic graphics commands to draw lines and turn through angles can easily be redefined into a microworld where the primitive drawing commands are to draw houses or streets or trees or whatever else makes for an interesting and manageable environment for the very young student.

In summary then, the power of LOGO does not reside in it being easy and quick to learn or even in the motivational value of being able to draw and produce music as well as manipulate words. These are clearly nice features to have but its true power is in its potential for building whatever type of learning environment the teacher feels is best suited to the needs of her specific students. LOGO's effectiveness in the classroom however is far from automatic. The successful use of LOGO in the classroom depends critically on the creativity and imagination of the teacher. Finally, although LOGO is the most publicized example of a programming language that is designed specifically to provide a procedural learning experience, other languages like BASIC could in principle be used in much the same way. They are not quite so good however since they are not quite as motivating particularly for young students and they require somewhat more work on the part of the teacher to use.

Initially, it might be assumed that the procedural learning potential associated with learning to program would not be particularly relevant to special education or at least be restricted to the above average or gifted students who seem to thrive in open ended, unstructured exploratory problem solving environments. Isn't it reasonable to assume that students at the other end of the intellectual continuum would not have the basic cognitive abilities to master programming let alone take advantage of the programming experience to acquire some procedural learning skills? Students with specific learning disabilities in particular have very short attention spans and lack the ability to attend to detail. Surely these students would be especially poor candidates for a programming course.

Such assumptions, while superficially quite plausible, turn out to be quite wrong. Some of the work for example done by Lindsay and Marini (1982) using computers with students who have severe and long standing specific learning disabilities suggests the opposite is true. [1]

According to Lindsay and Marini's work, not only are these students capable of becoming quite proficient programmers in both BASIC or LOGO, they seem to enjoy the activity enormously! Whatever lack of attention span or concentration they show in other contexts, they certainly don't show it when

---

1. The category of "specific learning disabilities" is primarily associated with someone who is of average or above intelligence but has a great deal of difficulty acquiring proficiency in oral and written language.

they are working on the computer.

Perhaps this ability to learn to program is not so surprising as it first appears to be. Most programming languages including BASIC and LOGO have very restricted vocabularies, relatively simple rules for making up new words in the language and a straightforward uncomplicated syntax. While programming languages do require rigid adherence to the rules, the programmer gets instant feedback in the form of a "?" or a "SYNTAX ERROR" from the computer if he or she makes a mistake in writing out the commands. If the error is in the logic or "semantics" of the program, then the computer will produce the wrong answer or perhaps no answer at all.

In these terms then, learning to program is really just another language learning experience for the exceptional learner but there is a difference. Like natural languages, a computer language has the motivational value of being functional. Once it is mastered, it can then be used instrumentally to obtain student defined goals. Unlike natural languages, programming languages are simple, explicit and precise. Providing a learning disabled student with the opportunity to learn to program may provide, perhaps for the first time, an opportunity for a successful and rewarding language learning experience.

Tools for manipulating language

Potentially the most powerful of the applications of computers in the schools today is the provision of computer based "cognitive tools". Because these programs are designed to augment our intellectual abilities, they are particularly relevant to students who have cognitive or communication disabilities. There is an important issue with regard to these tools as to whether they augment our cognitive processes or actually change the way we carry out a cognitive task. It is an interesting debate but it is beyond the scope of the present paper (see Pea, 1985, for an excellent and relatively recent summary of the current status of thinking on this issue). There is little debate however about the type of learning that is associated with the use of these tools --it is clearly procedural learning.

Of the cognitive tools currently available for teachers working with exceptional students, those associated with language and text manipulation are certainly the best developed and most widely available. Within the category of language tools, wordprocessors are the prime example.

Wordprocessors: The first generation

Certainly the most prominent and popular tool used for language manipulation in the classroom is the wordprocessor. Every modern microcomputer comes with a wordprocessing program of some sort and many have several such programs to choose from. Moreover, the wordprocessing programs are usually among the most powerful and sophisticated of the programs that a manufacturer has to offer.

These business oriented wordprocessors however are designed primarily as a tool for the secretary to use. Thus their main function is to facilitate the transcription of previously written or dictated text into final printed form.

In spite of their business orientation, many educators realized their potential as a tool for the teaching of writing very shortly after they first appeared. Since gaining their initial entre into the classsroom, the use of wordprocessors has grown enormously. A recent survey by the Toronto Board of Education (1985) for example found that, of all the potential educational applications of computers, elementary school teachers feel that wordprocessing is by far the most promising and important. These same sentiments have been echoed by teachers across the province.

It is impossible to predict in detail just how the widespread use of wordprocessors in schools and presumably eventually in homes will ultimately effect the speed and way in which children learn to read and write in the future. Controlled studies in this area are essentially impossible to conduct. We do however have a large body of published anecdotal testimony from teachers who have tried using wordprocessors in their classrooms. For teachers working with regular students, their praise of the wordprocessor has been almost unanimously positive. For teachers working with students with specific learning difficulties, the reviews have been positively raves(see Lindsay & Marini, 1982 for example). In short, the evidence is now quite substantial that at least from a teacher's perspective, wordprocessors are a very beneficial tool for working with students who have different types of exceptional learning needs (Lindsay, 1983, Humphrey, Kleiman & Lindsay, 1978).

It is not very difficult to see why wordprocessors are such an effective tool for working with students who have had such difficulty mastering print in school. First they seem to find computers a highly motivating instrument to work with. Frequently the introduction of wordprocessors into the classroom leads even the most discouraged student to make one more attempt at mastering print. Second students find it much easier to transcribe their ideas into print when working with a wordprocessor. Recognizing and pressing a key is far easier for these students than trying to form legible letters by hand (Lindsay, Woodruff, Bryson & Joram, 1986; Bryson, Lindsay, Woodruff & Joram, 1986). Third, the appearance of the document is important. Usually it is for the first time with a computer that the exceptional student can produce a highly professional looking final product (Lindsay & Marini, 1982; Daiute, 1985). With some of the more sophisticated printing packages that are now

available, students are able to set up and print a wide range of very high quality products in many varying formats from invitations, to posters, to newspapers to books.

Learning disabled students are not only willing to write more on a wordprocessor than they do with paper and pencil (ECOO, 1985), but they also seem more willing to try to find and correct the errors in what they have written. This may be because with a wordprocessor, students can correct their errors without running the risk of making more in the process. The computer has the advantage of not changing material that is already correct when printing out a corrected version of a text. Nor will the students have to laboriously copy out the whole text again after the corrections have been made. For students for whom penmanship is so difficult and bad, recopying a text after a correction has been made is often a counterproductive and painful exercise, they most likely will end up with more errors than they started with.

Similar, though not nearly so dramatic effects of wordprocessors have been reported recently for students at the other end of the intellectual continuum, that is students who are particularly adept at writing. Lindsay and his co-workers (Lindsay, et al 1986; Bryson et al, 1986) have found that above average grade eight student writers produced texts judged to be of significantly higher quality and more creative when they were working with wordprocessors than when they were using paper and pencil. Moreover, an analysis of think-aloud protocols taken as the students composed and revised their texts showed that working on the computer resulted in a significantly greater concern with the higher level rhetorical aspects of composition such as audience appeal and overall text cohesion than when the talented writers worked with paper and pencil. None of these effects were found with the average grade eight classmates of the talented student writers.

In summary then, the research to date indicates that a wordprocessor is an extremely effective writing tool across a wide range of curriculum activities and student ability levels. In fact computer based wordprocessors are sufficiently useful and powerful as teaching assistants for working with exceptional learners that they in themselves justify the purchase of computers even if wordprocessing was the only program to be used.

Educational wordprocessors: What makes them different

As mentioned above, the first wordprocessors to be used in the classroom were essentially business oriented wordprocessors designed to make the secretaries' transcription task faster and easier. When it began to be realized that wordprocessors might be useful in the teaching of writing, the first wordprocessors designed specifically for school use began to appear. The most popular of these was the Banks Street Writer, produced by the Bank Street College of Education.

Safety nets: The Bank Street Writer introduced a number of features not usually found in the business wordprocessors of the day. One of these features was the extensive use of "safety nets" designed to protect the student from his or her own errors. One of these safety nets was to question the student before any significant change or deletion of material to make sure that what was going to happen was what the student intended to happen. This greatly reduced the chances of the student inadvertently losing his or her text. A second type of safety net was designed to allow the user to change his or her mind after the fact. Thus you could not only delete, you could "undelete". Similarly you can "moveback" after you move something.

The second basic design feature that was introduced in this first generation of educational wordprocessors was the extensive use of on screen prompts to inform the user what the permissable commands were at any given point. Once a command was selected, prompts appeared to coach the student in how to use it. This minimizes the necessity for extensive training or reference manuals and maximizes the support given the novice user.

This first generation of educational wordprocessors then were "educational" only in the sense that they were built on the premise that the primary user would be a tentative and perhaps timid novice student who is just learning how to write rather than a professional secretary who is using it to transcribe someone else's writing. They did not incorporate any assumptions about the nature of the writing process as such or how best to teach it. Wordprocessors that emphasized particular approaches to the task of writing didn't appear until the next generation of educational wordprocessors.

Second generation educational wordprocessors

Top down versus bottom up writing: One popular assumption about the best way to go about writing something is that one should first make an outline of the material then fill in text according to the outline. According to this assumption then, good writing is "top down" affair in which one should start with an outline first. This assumption has been incorporated in the design of two popular wordprocessing programs -- Thinktank by Living Videotext and Framework by Ashton-Tate. Both of these programs are designed to make it easy to develop and work with outlines at varying levels.

Not all educators however believe that doing outlines first is the best way to write. One of the most recent and exciting of the new educational wordprocessors is a program called **NOTECARDS** (Brown, 1984). This wordprocessor makes it particularly easy for the writer to record essentially random notes as they occur to him or her while reading or thinking about a topic then organize these "notecards" after the fact into whatever coherent structure seems appropriate. In fact the user can experiment with a variety of organizations and NOTECARDS provide various some very nice graphic aids to help him do this.

Thus NOTECARDS is designed to facilitate a "bottom up" approach to writing and in this sense is essentially the opposite to THINKTANK and FRAMEWORK. Unlike its two competitors, however, it will also support the writer who prefers to write top down.

Perhaps the most important lesson to be learned from these examples is that there probably is no one "best way" to approach the writing task. For mature writers, the way in which they feel they write best is highly idiosyncratic (Smith, 1984). What works best for one person may be the worst possible approach for another. Given this conclusion, it is clear that when the teacher or administrator comes to make a choice of a wordprocessor, they should try to find one that is equally adept at supporting the writer who wants to write top down as the one who wants to write bottom up. The most attractive feature of a microcomputer as an educational tool is its ability to be personalized to the needs of the individual. This critical feature should not be foiled by making too strong a commitment to a particular writing or teaching philosophy in the choice of one's software tools.

Proofreading assistance

In general, wordprocessors are designed to facilitate the transcription and production aspects of composing and revising texts. A number of other programs are also being developed as part of an increasingly large set of language tools that are primarily designed to help with proofreading the text after it has been produced. Although the first programs to appear in this class focused on spelling errors, there are programs now to analyze "stylistic errors" as well. As a group, they have particular relevance to the exceptional learner since finding errors is a special problem for these students.

Checking spelling: The most important and widely available example of proofreading programs are those that check the spelling in a document. Spelling checkers are now part of almost all new wordprocessing packages. They work simply by reading a document and flagging any words that are not in its dictionary. Many of these programs can be switched "on-line" so that they give an immediate warning as soon as you type in an incorrect word or they will check any particular word on request.

The most sophisticated of the spelling packages not only flags the words that are not found in its dictionary, but also suggests the words that are most likely to have been meant. These suggestions are based on an analysis of common types of spelling errors people make as well as what legitimate words sound most like the word that was typed in incorrectly. Many of these programs will also allow the addition of words to the dictionary so that the dictionary can be personalized to the user's idiosyncratic vocabulary. Finally, many of these programs will provide a printout of all of the errors found in a text. This latter feature is particularly useful to help teachers keep track of how students are doing.

Spelling checkers are extraordinarily useful tools for teachers working with learning disabled students. These students are extremely prone to making spelling mistakes. The program to check spelling can go a long way towards taking much of the drudgery out of constantly having to check the student's work. The program can make the first pass through the work and pick up the majority of mistakes before the teacher has to get involved. It is important to note however that the program is far from foolproof. The teacher still must do a final check. Since the program doesn't know the meaning of the words, it will accept any word that is in its dictionary regardless of the context.

The spell checking program does make it possible for the students to become much more independent and responsible for finding and correcting their own errors. Moreover, since the program doesn't actually correct spelling for the student, the program can be used as an instructional aid. For each spelling error in her text, the student must identify which of the alternatives presented is the correct spelling or that none of them was the one intended.

Checking grammar: Spelling programs are currently the most widely used proofreading programs but some initial work is also being done on programs that can automatically identify grammar errors. The problem of detecting grammar errors is much more complex and hence these programs are far from being ready for mass distribution into the schools. When such programs do become available however, one can anticipate that, like the program for checking spelling, grammar checkers will assist students in becoming still more independent in being able to produce a correct document largely on their own.

Style analysis: Finally, in this general category of programs designed to help the writer analyze his or her text, the most sophisticated and complete set of program that have been developed to date go under the collective heading of Writer's Workbench (Frase, 1986). In addition to the standard things like checking the spelling, Writers Workbench programs identify and report statistics on more than thirty different "stylistic" features in a text. Overworked and ambiguous words, empty expressions, the number of colloquial expressions, unnecessary wordiness, the degree on concreteness (and therefore presumably clarity) of the writing are some of the many text features that are tabulated and reported.

Although without question, Writer's Workbench is the most ambitious "proofreading" program currently available, in terms of a demonstrated impact on improving writing, the verdict is not yet in (Keifer & Smith, 1983). It was originally developed at Bell Laboratories to help train commercial technical writers write more clearly and it seems to work well in terms of that objective. Its value for genres other than descriptive technical writing and its potential for the novice or the problem writer however has not yet been established empirically.

In principle, looking for mechanical errors or surface errors after drafting a text is an important part of learning to write clearly and well. Programs that provide help in this proofreading phase therefore should be useful in the classroom. Moreover they should be a particularly valuable aid to the exceptional learner who has so much trouble with the mechanical aspects of

writing.

It is important to keep in mind that the programs developed so far focus exclusively on analyzing the superficial aspects of a text. Essentially they can only deal with what can be analyzed without knowing what the words mean. This surface level of analysis is fine for the novice or the problem writer since mechanical correctness is something they have trouble with and can use help on. Such programs however have much less utility when it comes to supporting the more advanced or talented writer. These students need help with learning how to make "meaningful" rather than mechanical revisions in the text. Computer tools to help instruct students in these higher level revision strategies are not even on the drawing boards yet let alone ready for distribution to the schools. file: finknowl

Tools for manipulating knowledge

A relatively new cognitive tool has appeared on the educational scene that seems to have the potential for providing a very powerful procedural learning environment for students. This newcomer is the database management program. Although database programs have not as yet achieved the popularity or acceptance as wordprocessors, in the long run these programs have the potential of becoming even more widely applicable as a tool to support procedural learning in the classroom.

Database management programs can in principle be used to structure, organize and record any body of knowledge. To use such a program with a particular body of knowledge, the students or teachers must first define the characteristics of that knowledge. This involves deciding on the basic units or "records" of information that make up the knowledge base (e.g. a particular student's record in a student database, a vocabulary item in a lexical database) then on the specific features or characteristics that is to be associated with each unit of information (the name, address, and grades of a student; the part of speech, possible endings, subset, superset, and semantic meaning of a lexical item). After the basic unit of information and its characteristics have been decided upon, information can be entered into the database by recording the characteristics of each unit of information that is entered. Once the information has been recorded, any item or set of items can be searched for and retrieved by specifying the values of the characteristics of the sought after items. What the database program does is provide simple ways of defining the basic structure of the data then subsequently retrieving and tabulating the information in that database.

Teachers have reported using data base programs in classrooms to help students think about, organize and investigate the relationships among the information items in a wide variety of different knowledge domains. Sometimes it is used primarily as a fun activity such as having students try to define a database that would be appropriate for a "dating service". To define such a database, it is necessary to determine the most important type of information you have to know in order to match up boys and girls that will like each

other. This activity can lead to an interesting exercise in sociology. Other teachers have used database programs as part of a science project to track and tabulate the changing patterns of nature during fall and spring. Still others have used a database program to encode and teach the relationships among roots, prefixes and suffixes in a set of words. In addition to their educational uses in the classroom, database management programs can of course also be used as a classroom management tool by the teacher.

In terms if its application to special student populations, once again the first choice would probably for working with gifted students. For intellectually superior students, database management programs offer a wealth of potential challenging opportunities for thinking about, encoding and working with information in a variety of new and different ways. Precoded databases such as the Colliers Encyclopedia are now available in machine readable form for under $300. They might even go on, if finances permitted to explore some of the many commercial databases that are now accessible to anyone at quite modest rates (e.g. $10 per month). Commercial databases usually have electronic mail facilities associated with them so yet another new and exciting experience can be explored.

Would cognitively impaired students be able to acquire the skills necessary to use these new information manipulation tools. Databases are still relatively new to the classroom so we don't yet have the empirical data we need to answer this question unequivocally. If the their ability to use wordprocessors and master programming are good predictive guides however, then there are considerable grounds for optimism. If it turns out that they can master the basics of database systems, then they will have acquired access to one more potential cognitive prothesis that can help to normalize their school experiences and ultimately to increase their chances of leading a productive, independent life.

Mirroring the mind

The final class of computer tools for structuring learning environments to be considered is too new to have an agreed upon name yet. Generally goes under such terms as teaching metacognitive skills or procedural knowledge or sometimes by the rather cumbersome label of "computer supported intentional learning environments" (or CSILEs for short) (Scardamalia, 1986). The main idea behind this work is to try to develop techniques to teach students how to reflect upon and begin to understand their own cognitive processes. One thing that is needed is some way of holding up a mirror to the student's minds as they are doing some cognitive task in order to capture their problem solving activities. By externalizing the student's cognitive activity, their thought processes themselves can become an object of study that can be reflected upon after the task is complete. One question for current developers then is how can the computer be used to help externalize the student's thinking behaviour?

There is a particularly good working prototype of such a system which captures enough of the basic ideas that a review of this system should help clarify both the intent and the potential of this approach. For a change, the content area is algebra not language. The topic is solving differential equations in algebra. One traditional way of teaching this topic is for the teacher to present the equation to be solved, then suggest some operators that the student should try. The operators might be things such as "Try dividing through by a common factor" or "Try simplifying the expression on the left". The student's job is largely mechanical to do the work associated with applying the operator. With this approach the teacher is doing the high level thinking work of deciding on the most appropriate operator to apply and when to apply it. The student is doing the mechanical work of carrying out the arithmetic operations.

An ingenious program called **AlgebraLand** (Brown,1984b), that incorporates many of the features of a fledging CSILE, has been developed at the Xerox Parc Palo Alto research centre to teach grade 10 algebra students. The program poses a problem to the student [e.g., 4(2 + N)=20] and displays the various operators that the student can choose from. When the student chooses an operator (by pointing at it with a mouse), the computer will carry out the necessary arithmetic and display the equation after the operator has been applied. The student can then choose another operator and continue until he or she has solved the equation or given up.

Thus one thing AlgebraLand does is to act as a computational tool that frees the student to concentrate on the higher level aspects of problem solving. A second thing it does is to build up a complete graphic record of the various solution paths the student takes as he or she attempts to solve the problem. This is how it acts as a mirror and produces an external record of the student's problem solving behaviour that can serve as data that the student can later inspect and evaluate.

Note that in AlgebraLand, the role of the student and the teacher has been essentially reversed from that of the live teaching situation discussed earlier. The student does the former teacher's job of deciding on which operator to apply and when to apply it while the computer does the student's former job of carrying out the mechanical arithmetic associated with applying the operators.

Though still a prototype, AlgebraLand has some very interesting features that are worth considering. It structures the situation so that the student can focus on the more creative aspects of a problem solving situation rather than on the mechanical aspects. It provides a graphic picture of the student's hypothesis testing activity during the problem solving episode that the student can compare with that of an expert solving the same problem or with herself solving different problems on different occasions.

In spite of its many positive features, AlgebraLand is somewhat deceptive as a general model. One of the reasons it works so well is that it is dealing with a very explicit problem area where both the possible operators and the solution paths are limited and well defined. Could the same technique be applied to fuzzier less well defined domains like writing? Could the provision of automatic spelling correction be considered equivalent to

carrying out error-free arithmetic for the algebra student? In both cases, the computer is assuming some of the more mechanical aspects of a complex task. But what are the set of operators from among which the writer chooses as he goes about composing and revising her text? Can they be listed on a menu and automatically carried out on any piece of text by the computer at the push of a button? Obviously there are issues to be worked out in trying to generalize the ideas of AlgebraLand to other problem domains.

Nevertheless, AlgebraLand is a working example of a cognitive mirror and therefore is instructive as a prototype. It makes some rather vague ideas concrete. It also provides a working environment with which to experiment. Though it has yet to be tested, it may very well be for example that when freed from the burden of carrying out their highly error prone arithmetic, the learning disabled student looks much more like his normally performing peers than he does when his mechanical errors are there to confuse the comparison. How about testing the mathematical gifted on AlgebraLand? Would the "problem behaviour graph" showing the path to solution have any unique characteristics for a student who was particularly gifted in solving algebra equations? Could the principles incorporated in AlgebraLand be generalized to provide a computer based prothesis where the computer could do the low level mechanical tasks under the direction of the exceptional student and hence compensate for some of the student's deficits?

At the moment the number of concrete examples of systems that act like cognitive mirrors are too few to be able to predict what the range of applicability is likely to be or even if the ideas are valuable to have any long term staying power.

PROGRAMS FOR AUGMENTING THE SENSES

To this point, we have been dealing exclusively with the sue of computers to support the cognitive activities of exceptional students. Students with cognitive exceptionalities represent by far the largest group of Special Education students that school boards have to deal with. A much smaller number of students have difficulty learning in normal school situations not because of cognitive impairments but because they are sensorially impaired in some way.

As is the case in the cognitive domain, computers have a unique role as a potential sensory prostheses for the students who have either receptive impairments (e.g., the blind and the deaf) or expressive impairments (e.g., the student who is unable to speak). This latter group present a particularly poignant problem since their inability to communicate means that they are usually totally cut off socially from their age matched peers.

In principle, students with sensory or productive problems can profit as much from the cognitive tools discussed above as do the cognitively imparied

students. They have an added problem. They may have to have a special "interface" in order to gain access to the tools. Fortunately a number of computer programs are available to augment the input or output of information.

Computers for students who can't hear

Regardless of whether the basic educational philosophy of a school jurisdiction is teach the deaf student to sign or to try to maximize the student's oral communication, all programs for the deaf attempt to help the student learn to produce as much understandable speech as he or she is capable of. Recently a number of specialized computer programs have been developed to help the deaf learn to produce understandable speech. In these programs, the computer monitors and displays a visual representation of the acoustic target signal that has been produced by the trainer. The child repeatedly speaks into a microphone trying to reproduce the same pattern she sees on the screen. When the child's utterance exactly matches the target utterance it will sound just as understandable. Since the child can proceed at her own pace and take as long as she likes to acheive the match, we have yet another example of the computer's ability to free the handicapped individual from a dependence on others, this time in the sensory domain.

Sometimes the computer turns out to provide a surprisingly good solution to a difficult problem that arises. An illustrative case of this involves a much older student than we have been considering so far. The student was enrolled in the Master's program after having been totally deafened and partially blinded by an operation to remove a brain tumour in his final undergraduate year. Since he could not hear and had only limited tunnel vision in one eye, taking classes was obviously a problem for him. The solution was remarkably simple -- a portable computer, a wordprocessor, and a notetaker. The notetaker sat in class with the student using the wordprocessor (Bank Street Writer) to transcribe the class lecture and discussion as it went along. The student was able to read from the screen and therefore participate in the class. As an added bonus, the student had a complete set of typed lecture notes that could be corrected by the instructor immediately after class.

Though it was not necessary for the student in this particular case, the text could easily have been displayed and printed in any print size if the student's vision had been poorer. Finally, the computer was also a helpful tool for integrating the student socially. It became his portable notebook that he could use for talking with people he met in the halls or at parties. One difference in these social interactions was that the computer maintained a permanent record of his conversations.

Computers for the visually impaired

In addition to specialized programs for the deaf there are also specialized programs for visually impaired students. The simplest solution for these students if their impairment is not too severe is to use one of the large print "utilities" that can be added to any program so that the screen or printer output is large enough for most visually impaired students to read. In some cases these utilities are built right into the wordprocessor. Print size becomes just one more option you select before you begin to write.

For those with no vision at all, there are "automatic brailers" that the student can use to transcribe notes in class into braille for later use. If the student does not want to learn braille, there are also very powerful but unfortunately, correspondingly expensive (i.e. $20,000) computer based reading machines that can transcribe print directly into a voice output. This allows the blind person to read any material he or she chooses without have it taped or brailled beforehand.

Once again, the same theme is emerging in yet another context. The the introduction of a computer has the potential for enormously increasing the range of autonomous action for the handicapped individual. With an automatic reader, even the totally blind person can keep up with the best sellers if he or she chooses.

Computers for students with no voice

The final category of programs for augmenting the senses is perhaps the most exciting of the new developments in the general area of computer based prothesis. This is the work on computer generated speech which, for the first timeis giving a voice interaction capability to nonspeaking individuals.

A significant number of students are born unable to use their vocal apparatus to communicate due to a variety of crippling diseases, the most common being cerebral palsy. Because it is so difficult to establish any sort of communication with these children, it is hard to know the extent of cognitive impairment, if any, that they may have. Also, because they can't communicate, normal social and emotional development is totally disrupted. Relative to other exceptionalities, this group of children numerically is not particularly large (e.g. according to a survey conducted in the Fall of 1985, there are about 2300 school aged students currently in the educational system in Ontario). Relative to other children however their impairment is extremely severe and their eucational and social prognosis are among the poorest of all exceptional children.

For the first time, sophisticated computer technology offers some hope of being able to give these children a voice and hence a chance at some semblence of normal life. Even the child who has the barest minimum of response capabilities, perhaps only the ability to blink an eye or to flex a forehead muscle, can be given a voice with a computer. If these minuscule responses are sufficiently reliable, they can be sensed by a computer and converted into signal that can in turn can be used by a computer based language system to construct any message the child wants to communicate. Once constructed, the message can be spoken or printed on the child's command.

Again the computer holds the promise of increased independence for the handicapped child. For the nonspeaking individual, however, the computer is far more than an educational aid. It may be the child's major lifeline to the outside world.

REFERENCES

Brown, J. S. (1984a) Idea amplifiers: New kinds of electronic learning environments. Palo Alto, CA: Xerox Palo Alto Research Center, Intelligent Systems Laboratory

Brown, J. S. (1984b) Process versus product: A perspective on tools for communal and informal electronic learning. In S. Newman & E. Poor (Eds.), Report from the learning lab: Education in the electronic age (pp 41-57). New York: Educational Boradcasting Corporation.

Brown, J.S., Burton,R.,& deKleer,J.(1982) Pedagogical, natural language, and knowedge engineering techniques in SOPHIEI,II, and III. In D. Sleeman & J.S. Brown(Eds.), Intelligent tutoring systems (pp. 230-282). New York: Academic Press.

Bryson, M., Lindsay, P.H., Woodruff E., & Joram E. (1986) Augmented wordprocessing: The influence of task characteristics and mode of production on writers' cognition. AERA Annual meeting, San Francisco (manuscript)

Cameron, J., Kress, F. (1984) Wordprocessing and the learning disabled high school student. Toronto: ECOO Output 5(2), 28-30

Frase, L. T. (1986) Creating intelligent environments for computer use in writing. AERA Annual Conference. San Francisco (manuscript)

Goldenberg, E. P., Russell, S.J., Carter, C.J. (1984) Computers, education and special needs. Reading: Addison-Wesley

Hagen, D. (1984) Microcomputer resource for special education. Reston: Reston (Prentice-Hall)

Hayes-Roth, B., & Thorndyke, P. W., (1985). Paradigms for intelligent systems. Educational Psychologist, 20, 231-241.

Hollan, J.D., Hutchins, E., & Weitzman, L. (1984). STEAMER: An interactive inspectible simulation-based training suystem. AI Magazine, 5, 15-27

Keifer, K. E. & Smith, C. R. (1983) Textual analysis with computers: Tests of Bell Laboratories' computer software. Research on the Teaching of Writing, 17(3), 201-214

Kleiman,G. Humphrey, M & Lindsay P. H. (1980) Microcomputers and hyperactive children. Ontario mathematics Gazette, 10(2), 30-32

Lindsay, P. H., & Marini. A. (1982) Microcomputers and exceptional children: Releasing the potential. Final Report of the Trillium School Project

Lindsay, P. H. (1982) Microcomputers in Special Education - Do we have any choice? In Ontario Teachers' Federation's Special Needs/Special Help,19-21

Lindsay, P.H., Woodruff, E., Bryson, M., & Joram, E. 1986 Writing with computers: Its impact on composition and revision in talented and average students. University of Pittsburg Conference on Writing and Computers (manuscript)

Papert, S. (1980) Mindstorms: Children, computers and powerful ideas. New York: Basic Books, Inc.

Pea, Roy D., (1985). Beyond amplification: Using computers to reorganize mental functionaling. Educational Psychologist, 20, 167-182.

Scardamalia, M. (1985) Computer supported intentional learning environments. Center for Applied Cognitive Science (OISE)

Smith F. (1982) Writing and the writer. New York: Holt

Toronto Board of Education Survey (1985)